Paper submitted within the scope of the Master's Thesis
Master of Industrial Sciences
GROUP T – Leuven Engineering College – 2008-2009

Behavioural measurement software tool with hardware component

Li Xiran*, Shan Kedi*, Zaman Bieke†, Van Overveldt Maarten‡

*Master student E-Media, GROUP T – Leuven Engineering College, Vesaliusstraat 13, 3000 Leuven
†Faculteit Sociale Wetenschappen, Studentenadministratie, K.U.Leuven, Parkstraat 45 bus 3600, 3000 Leuven
‡Unit Information, GROUP T – Leuven Engineering College, Vesaliusstraat 13, 3000 Leuven, maarten.van.overveldt@groept.be

Abstract

In this Master Thesis, we introduced the Behavioural measurement software tool with a hardware component (Approach-Avoidance Meter) which is an application providing a measure function by showing screenshots of games on a computer screen and then recording users' reaction. We used the Scrum model, one of the Iterative and Incremental Development methods, to develop the application. We finished most of the requirements and provided it to our co-promoter. The key of the application is the reaction time which is the time that elapses between showing images and tester's response (pushing or pulling the joystick) to it. We tested the application and noticed that the application caused time delay errors which resulted in the inaccuracy of reaction time. Then we found solutions and solved the time delay error problem caused by the hardware scanning and signal transporting. The co-promoter also tested the application and gave us the feedback. According to this feedback, we improved the application. In this paper, we explained how the application was implemented according to the software structure chart.

Keywords: Approach-Avoidance Meter, Time delay error, Joystick, Reaction, Game, measurement

1. Introduction

1.1. Approach-Avoidance Meter [1]

How do you analyze how fun a game is? The Approach-Avoidance Meter, which is also a behavioural measurement software tool with a hardware component, is introduced as a tool which measures how fun a game for a player through the registration of upper body movements and automatic pull or push reaction with a joystick.

1.2. Upper body movements

First, the Approach-Avoidance Meter registers how long and how often the user leans forward or backward while playing a game. If possible the Approach-Avoidance Meter measures whether the user has a sunken chest or stretched/extended upper body. How this is measured, is free to choose (e.g. through sensor in garments). Leaning forward and having the upper body stretched are signals of interest in the game. Drawing back on the chair and having a sunken chest might suggest that the user may be experiencing rather negative emotions (such as boredom, frustration ...).

1.3. Joystick movements

Secondly, the Approach-Avoidance Meter also measures fun by registering joystick movements. Experiments have shown that people are faster to pull (versus push) the joystick towards oneself in response to the presence of positive stimuli on the screen and faster to push the joystick away from one self in response to the presence of objects (such as game) they do not like. The Approach-Avoidance Meter will analyze this by showing screenshots of games on a computer screen. The users will be given the task to react as fast as possible by pulling or pushing a joystick away or towards the screen.

In this context, we will principally only explain the joystick movement feature. We designed an Approach-Avoidance Meter with the possibility to measure the joystick movement.

Our Approach-Avoidance Meter will be used in usability and likeability tests of games with children. After having played a few games, the Approach-Avoidance Meter will present a sequence of predefined screenshots of the games on a computer screen. The child user will be asked to respond to the screenshots as fast as possible by pulling the joystick forwards or away from the screen.

2. Requirements

2.1 Tester

Record the tester's information (Name, Age, Gender).

The application should present a sequence of predefined screenshots of the different games. Each

screenshot remains on the screen until the child has moved the joystick.

The application should measure the reaction time and the direction of joystick movements for each screenshot. After test finished, all the data are saved into a database.

2.2 Research

Import the screenshot of games

Manage image list

Export the tests' data to an Excel file. The Excel file should includes FilePath (image path), Action (joystick movement), ActTime (ms), CreateDate

Figure 1 shows the user case diagram of tester terminal.

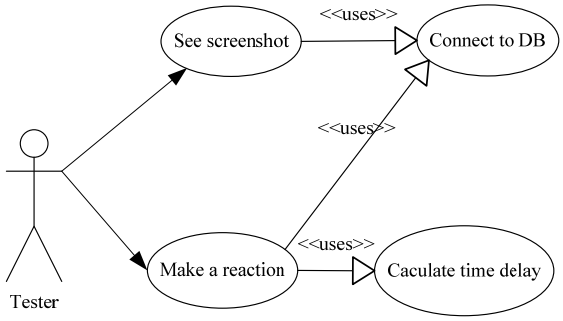


Figure 1. User case diagram of tester terminal

Figure 2 shows the user case diagram of researcher terminal.

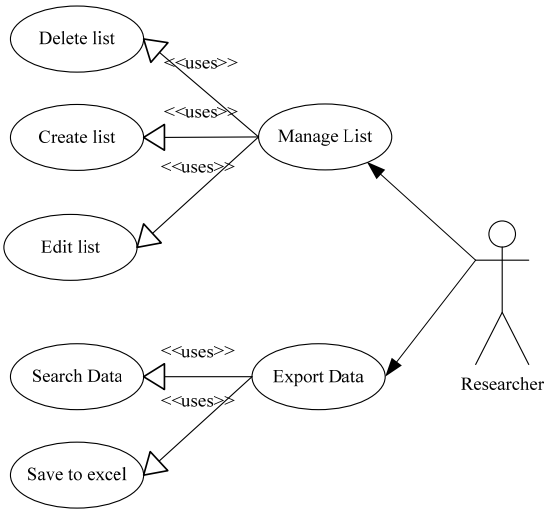


Figure 2. User case diagram of researcher terminal

3. Methods

3.1 Scrum model

Scrum is a software developing method that includes a set of practices and predefined roles. Scrum is an iterative incremental framework for managing a complex work (such as new product development) commonly used with agile software development. Although Scrum was intended for management of software development projects, it can be used to run software maintenance teams, or as a general project/program management approach. [2]

During each sprint, a 15-30 day period (length decided by the team), the team creates an increment of potential usable software. That means after each sprint, new features will be added. In each sprint the requirements that have the highest priority will be done. After each sprint, the team will hold a meeting. During this meeting the Product Owner informs the team of the goals he wants completed during the next sprint. The team then determines how much of this they can commit to complete during the next sprint. During the sprint, no one is able to change the goals, which means that the requirements are frozen for one sprint.

Figure 3 shows the Scrum process.

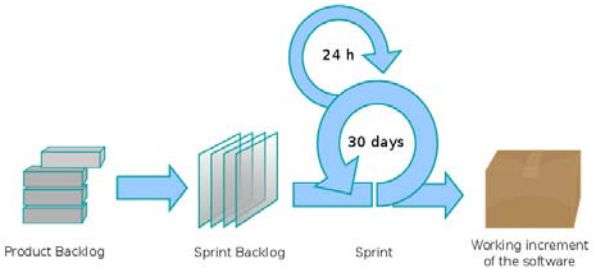


Figure 3. The Scrum process [2]

In our Master Thesis, we make each sprint shorter than usual, about 14 days.

Each day in the scrum, a project status meeting should be hold. Due to our course scheduler, we can not hold a meeting everyday. So we make it once in every two weeks. During each meeting, we should talk about following questions:

What have you done since yesterday?

What are you planning to do by tomorrow?

Do you have any problems preventing you from accomplishing your goal?

Both team members talk about the past sprint. The purpose of the meeting is to make continuous process improvement.

A key principle of Scrum is its recognition that during a project the customers can change their minds about what they want and need (often called requirements churn), and that unpredicted challenges cannot be easily addressed in a traditional predictive or planned manner. This is the most important reason that why we choose Scrum Model. Because the developing

period is quite short, we don't have time to make a good priority table and a good project plan. We have to develop and design concurrent. This will lead to our application's functionality simple. So we might extend the functionality after each meeting. So we need a flexible development model.

We will use OOP (Object Oriented Programming) framework to develop this application.

In our developing period, we have five sprints.

Sprint 1: developed core function which connect joystick and application and transfer data.

Sprint 2: based on the core function, we finished the whole testing process in this sprint, including: displaying images, calculating reaction time, saving testing data into database.

Sprint 3: as we have finished the whole testing process, we added the management component to filter, export testing data and manage the images.

Sprint 4: improvements of reaction time calculation and time delay error.

Sprint 5: we made the final application testing and did some improvements.

3.2 Sprint 1

In this sprint, we developed the core function of the application which connects the application and joystick.

We designed the joystick component. The development environment is C++ and windows. Figure 4 is the operation flow chart of receiving data from joystick.

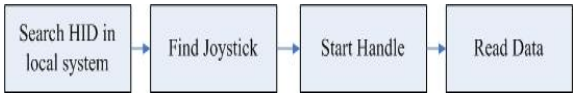


Figure 4. Flow chart of receiving data from joystick

Because the application was developed in java, so we exported the joystick component as a DLL.

3.3 Sprint 2

Based on the previous sprint, we made a timer to calculate the reaction time. As soon as the users see the picture, the timer will start. After the application receiving the valid data from the joystick, the timer will stop and calculate the reaction time.

Figure 5 is the data flow chart of a complete test.

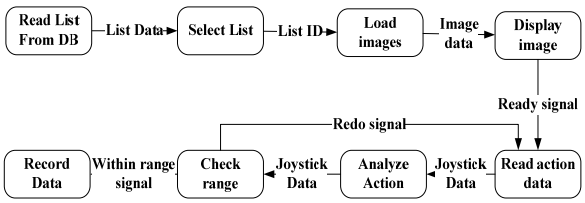


Figure 5. The data flow chart

After the application started, it shows all the lists which are set up by researchers. The users must select a list before starting test. The images will be loaded and shown to the users.

The researchers and testers will work on the same machine. So we can integrate applications and data (database, images) together. In the future, if we want to copy the application to another machine, we have to copy all the images to that machine too. If there are too many images, that will be a big trouble. So we designed to store all the images in the database. That may make the database very large. But the advantage is that all testers' machine can share one database. The researchers needn't to copy the images to every machine. The application can download the images data from the database server. When the application finishes downloading all the needed images data, it will generate temporal image files and save them on the local machine. That's because if the application downloads the image just before it is displayed, it will cost some time and this can affect the measurement test result.

When the users see the image, they will move the joystick. The signal from the joystick which is generated by user's action will be sent to the application. Then the signal will be recognized by the application. If the action's data is not valid, the application will drop it and receive a new one. As long as the action's data is in range, it will be saved in a DB.

Figure 6 is the structure of the terminal.

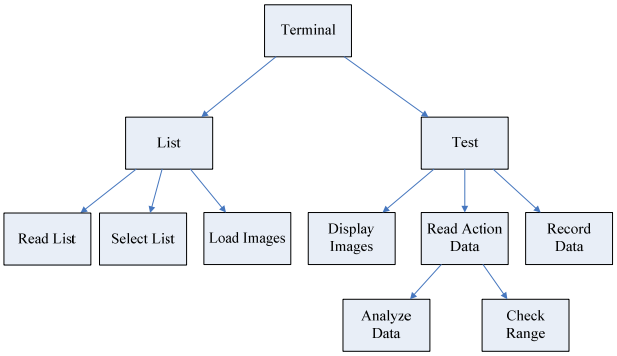


Figure 6. The structure of the terminal

We divide all the functions into two parts: List Part and Test Part.

3.3.1 List Part

List Part is used for selecting list and downloading images from server. How to save images? We have two methods.

The first one is to save all the images on the local hard disk and save the path of the image files into the database.

The second one is to save the images data into database directly. If one list is selected, it will

download all the images which belong to the list to the machine at once.

We compared these two methods. Finally, we chose the second one.

The reason is that the second method is easy for future extending. So, the database server and the application will be installed in the same machine. And there are not too many differences between the two methods. But future more, if we want to extend the system, add one or more test machines. How to transfer the images? If we use the first method, we need to copy all the images to each test machine. If there are hundreds of images, we can image that must be a hard work. But the second method can solve the problem. Once the researches want to add a new machine, he just needs to make the application connect to the database server. The application will download images automatically.

But it still has some weakness. This method demand high bandwidth. But most of the time the test will be taken at local network, so that the bandwidth will not be a problem.

After we save all the images on the database, when shall we download them? Here we also have two choices: downloading them before the test starting or download one picture after the previous picture is shown on the screen. We choose to download them before the test starting. Although it may take several minutes to download and generate file, it will not affect the test result. If the application downloads the images during the test, it may generate extra time delay error and affect the test result.

The following table shows the IPO (Input Process Output) diagram of the main function in List Part:

Table 1. Load Images

Input	Process	Output
List's ID	Find all the images belong to selected list and download all data from database.	After receiving all the data, generate temporal files on the local hard disk

3.3.2 Test Part

The main functions of Test Part are receiving data from the joystick and changing images every 6 seconds. To connect to the joystick and receive data, we need to use the DLL component which is developed in C++. It will return an integer that from -255 to 255, indicating that the joystick moves from forward to backward. After receiving the data, first we need to analyze the action in order to know the kids are pushing or pulling the joystick. Then we will check if the action is valid. Because the kids are curious, they may shake the joystick slightly. So we set a valid boundary. Only when the data exceeds the range, it will be recorded.

Following tables show some important functions' IPO diagram of Test Part.

Table 2. Read Action Data

Input	Process	Output
No input	Find Joystick Get the value of the stick's position	Position value

Table 3. Analyze Data and Check Range

Input	Process	Output
position value	If the value is smaller than -40, the action is push. If the value is bigger than 40, the action is pull. If the value is between 40 and -40, drop data and receive a new data.	action

3.3.3 User Interface Design

Because most of the users are kids, so we can not use complicated sentence and instructions. Basically, the interface should be simple and easy to understand.

For example, in order to let the kids know when they can stop moving the joystick, we must display some information on the screen. Instead of using text, we use a "finish" image to indicate that the users will wait to look at the next screenshot.

Figure 7. "Finish" indicator

The images which are shown on screen should be large enough. Here we set the size as 600px*400px.

3.3.4 Development Discussion

During the development, we found a critical bug. Once an image is shown on the screen, application will start a thread to connect to the joystick and read position data. As soon as the application received a valid data, the thread will be closed. But we found that if the kids do not move the joystick, the thread will always exist. Then after the next image is shown, another thread will be started. So there are two existing

threads at the same time. If the kids move joystick at this time, two position data will be returned. The application would be confused. It doesn't know which data should be recorded. The threads should match the images. So we save the image's ID in thread. Only when the image's ID saved in the thread match the current used image's ID, the position data will be received and recorded.

3.4 Sprint 3

At sprint 2, we finished the whole process of the testing. So in this sprint, we added all the management functions, including image management and testing data management.

In figure 8 the data flow chart of management functions is showed.

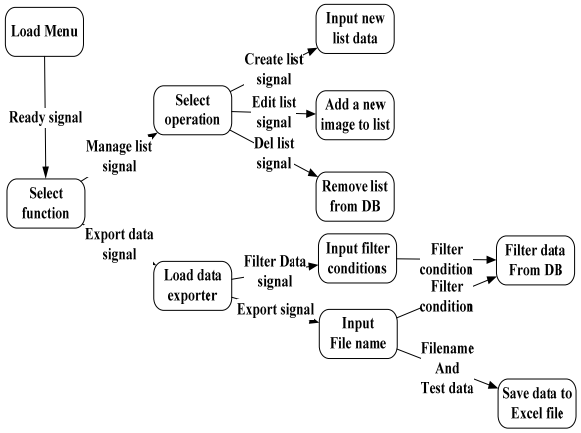


Figure 8. The data flow chart of management functions

After the researcher started the application, a main menu screen will be shown. The researcher can choose the menu to manage a list or export data. By choosing the "list" button, the list manager will be loaded. Then researcher will have three choices: create, modify and delete.

Clicking the "Create" button will enter the list creating screen. When a new list is been created, the researcher must input the list name and specify a folder which contain images. The application can automatically upload all the images file contained in the folder. If researchers want to add or remove an image from a list, he can click the "Edit" button. At the Edit screen, researchers can select an image to remove or click a button to add a new image. After finishing all tests, researchers can delete the list from the database.

The application also allows researcher to export data to Excel files. But first they need to filter data first. At the data manager screen, they can select a tester from a window. Then the application will display the entire tester's testing data on the screen. Then the researcher can click the button to choose a file name for the Excel file. Afterwards the file will be generated.

Figure 9 shows the structure of the terminal.

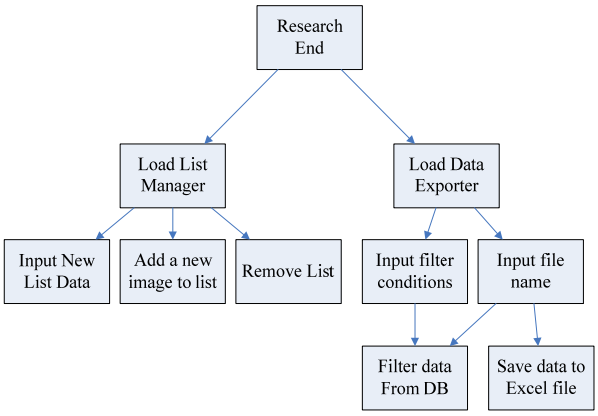


Figure 9. The structure of the terminal

There are also two parts in researcher terminal: List Manager and Data Exporter.

3.4.1 List Manager

List manager provides three functions: create new list, edit existing lists and delete the lists. When a list is created, the researcher can save all the images in a folder. The application will upload images in the folder. Because the application should present a sequence of predefined screenshots for the tester, the order of the images in the list depends on the order in folder. After the list is created, the sequence can not be changed. New added image will be at the end of the sequence.

Table 4. IPO diagram of Creating List

Input	Process	Output
The name of the new list. The absolute path of the selected folder	Find the images in the folder. Upload all the images to the database.	Display the new list's name on the screen

3.4.2 Data Exporter

At the data filter screen, the researcher should create the tester's name as condition. Our design method is that the researcher can't create the name but select it from a list. If there are too many testers, the name list is maybe very long. But it is still more convenient than that the researchers can create the names by themselves.

To export the data into Excel file, we use following statement:

select pictures.FilePath, testdata.Action, testdata.ActTime, testdata.CreateDate INTO [Excel 8.0;database="+savepath+".xls].[sheet1] from testdata

inner join pictures on pictures.PicID = testdata.FilePath where UserID = '"+userid+"'

Table 5. IPO diagram of Filtering Data

Input	Process	Output
Tester's name	Search the database, and find all the testers' data.	Display on the screen

Table 6. IPO diagram of Exporting Data

Input	Process	Output
File name Tester's name	Search the database, and find all the testers' data. Create an Excel file with the given name. Save all the data into the file.	Store the data into a Excel

3.5 Sprint 4

We finished the demo of the Approach-Avoidance Meter application. But the most important function of this application is to provide the reaction time, which still had some errors and was not exact. So we did some improvements of the reaction time calculation and time delay error so that there were no time delay errors present anymore.

3.5.1. Reaction time

Next formula shows how the reaction time is calculated:

Reaction time = End time - Starting time;

Starting time: The current time is recorded when the screenshot start is showed to the tester.

End time: The current time is recorded when the tester start to react as fast as possible by pulling the joystick forwards or away from the screen.

3.5.2. Time delay errors

There are two reasons that may cause the time delay errors: Hardware signal transporting and Application running time.

3.5.2.1. Analysis of the time delay errors

Hardware signal transporting:

We know that the time delay about the digital signal of joystick movement is generated by digital memory buffers on the joystick. Also the PC memory needs to spend some time to process the signal for the application program. So the time delay in hardware is the combination of both of them.

Application running time:

When the electrical signal is received by the application program, the time delay error is the application running time from the software. And this time delay error is decided by the time complexity of the program and the implementation process of the application tools.

Also there are some other time consumptions, which include the time of the game image loading, the time of the data transmission to the database, etc. But we have no need to worry about those, which don't influence the calculation process of the reaction time. Also the applications of them are never called in the runtime of calculating the reaction time.

3.5.2.2. Calculation of the time delay errors

How to calculate and eliminate time delay error is the most important.

Time delay error (Run-time process) = End of the process time - Signal reception time;

Signal reception time: the current time is recorded when the application program receives the electrical signal which is generated by the joystick.

End of the process time: the current time is recorded when the program process of the reaction time calculation is finished.

Because we use the program language Java as the development tool, we use the System.currentTimeMillis Method of Java to get the current time in Java programs.

System.currentTimeMillis Method: Retrieves the current time in milliseconds; [3]

Also we use the program language C++ which is used in the driver program of the joystick, so we use the _ftime Function to get the current time in C++ programs.

_ftime Function: The '_ftime' function sets the time and 'millitm' members of the 'timeb' structure pointed to by 'timeptr' to contain the seconds and milliseconds portions, respectively, of the current time in seconds since 00:00:00 UTC (Coordinated Universal Time), January 1, 1970. [4]

So, Time delay errors **(**Run-time process**) =** End of the process time **(**System.currentTimeMillis Method in Java**) -** Signal reception time **(**_ftime Function in C++**)**.

So, figure 10 shows the flow chart of calculating the reaction time and time delays in the program.

The DLL is the connector which connects the Joystick and returns the position of the Joystick. It is programmed in C++, and is the dynamic link library in Java.

Connecting: Scan USB port and connect to joystick.

Signaling: get the joystick's position and send to DLL.

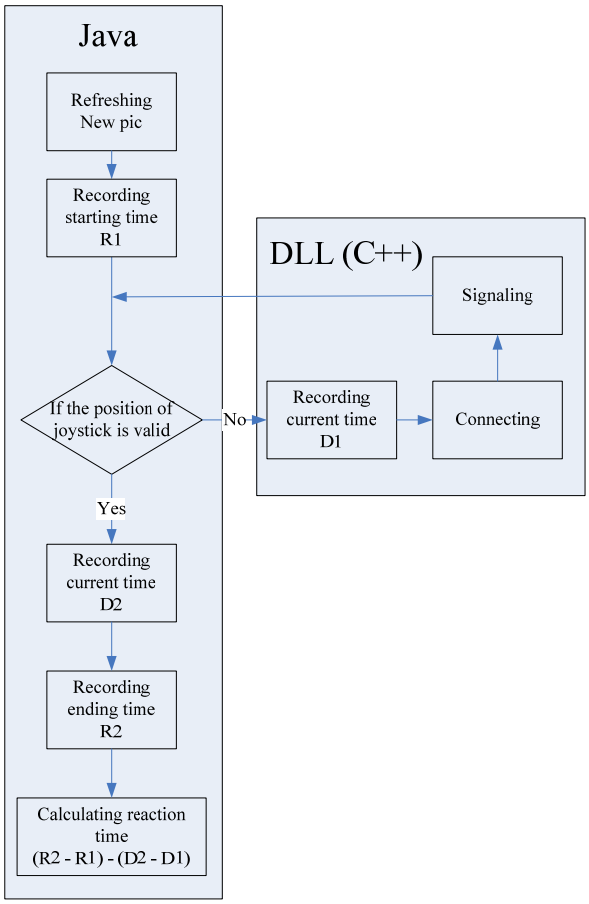


Figure 10. The flow chart of calculating the reaction time and time delays in the program

3.6 Sprint 5

We made a final application testing in the sprint and we got some feedback:

The size of displaying images is too small.

Add some simple instructions on the interface.

After downloading all the images, the application should have a count down timer to tell the kids that the test will start in 10 seconds.

We then improved the application based on this feedback.

3.6.1. Time delay error test

3.6.1.1. Reaction time theory

According to Hick's law, choice reaction time increases in proportion to the logarithm of the number of response alternatives. The law is usually expressed by the formula $RT = a + b\log_2(n + 1)$, where a and b are constants representing the intercept and slope of the function, and n is the number of alternatives. [5]

And mean RT for young adults is approximately 215 milliseconds to a detected visual stimulus. [6]

3.6.1.2. Time delay error test research

We chose a PC with the environment:

CPU: Intel(R) Core(TM)2 T7200 2.00 GHz

Memory: 2.00 GB

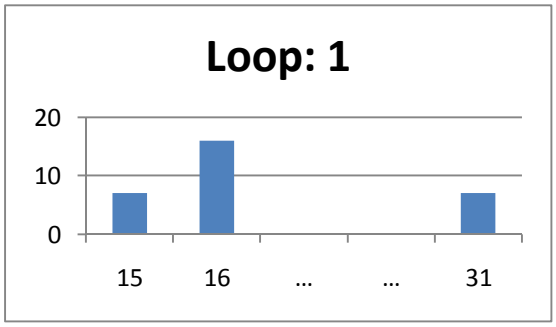
Operating System: Microsoft Windows XP Professional

Loop: 1 Test: 30

Samples (ms):

16/15/31/16/16/15/16/31/16/15/16/31/16/16/15/16/31/16/15/16/31/16/15/16/16/31/16/15/16/31/

Table 7. Distribution: (16±5 ms)

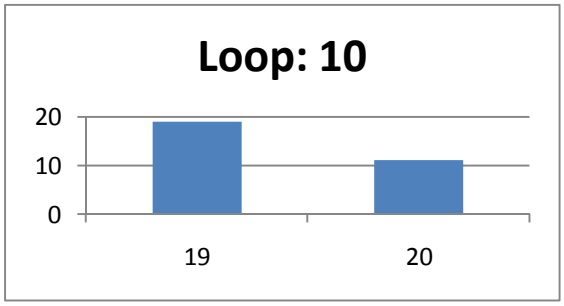


Loop: 10 Test: 30

Samples (ms):

188/203/187/188/187/188/203/187/188/187/203/203/188/203/188/187/188/203/187/188/203/187/188/203/203/188/187/203/188/203/

Table 8. Distribution: (19±1 ms)

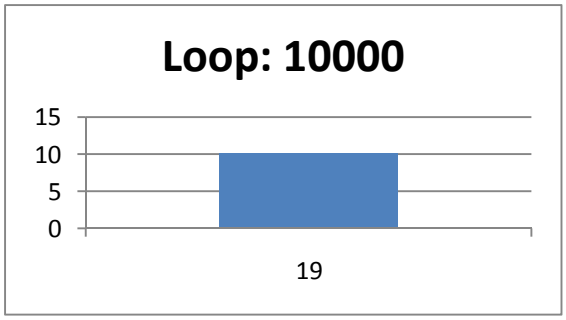


Loop: 10000 Test: 10

Samples (ms):

193500/194703/191906/194390/193719/192922/192828/193109/193415/192914

Table 9. Distribution: (19 ms)



We know that the more loops, the higher accuracy.

So we get the conclusion that the delay time is 19 milliseconds in this environment.

3.6.1.3. Time delay error discussion

We know that the time delay error of 19 milliseconds which is almost the same then a mean RT of 215 milliseconds is big enough. It can influence the measure results. So we eliminate time delay error for each measure. But for the accuracy of the Approach-Avoidance Meter research we must know the data of the time delay error.

And the 19 milliseconds include the delay time that the program is postponed and the delay time that the program identifies the joy stick signal.

But there is another delay time we didn't calculate, which is the time that the signal from the joystick to the interface of the computer hardware.

Also there is an important key. We know different computers and different state with the current computer generate different delay time. So we should calculate the delay time according to the experiment computer.

4. Result

4.1 Application Data Structure

In this application, we need to save users' data, images' data, lists' data and tests' data.

Table 10. List

Name	Type	Instruction
ListID	String	The unique ID for each list
ListName	String	Input when created. Be shown on the list selector.
CreateDate	datatime	Record the data when the list is created.

Table 11. Images

Column	Type	Instruction
ImageID	String	The unique ID for each image
ListID	String	A list's ID which the current image belongs to.
ImageName	String	The image's file name.
ImageData	Binary Data	The image's content will be translate to binary code and saved.
CreateDate	datatime	

Table 12. Users

Column	Type	Instruction
Username	String	Tester's name
Gender	String	Tester's gender
Age	int	Tester's age
CreateDate	datatime	

Table 13. TestData

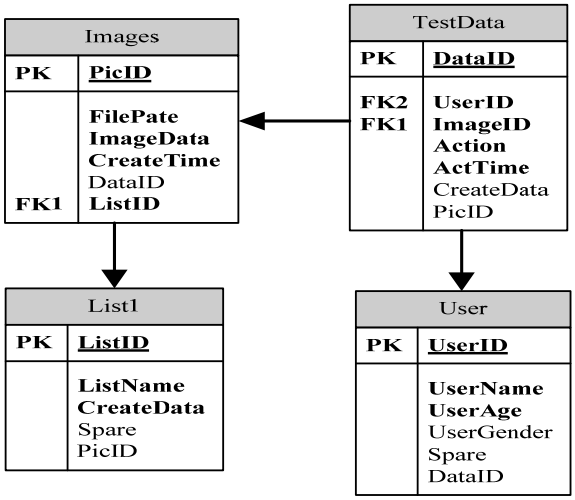
Column	Type	Instruction
DataID	String	The unique ID for each data
UserID	String	tester's ID
ImageID	String	Image's ID
Action	String	The joystick's movement, pull or push.
TimeDelay	int	The reaction time.
CreateDate	datatime	

Table 14. shows the relationship between data structure and testers' functions.

	List	Images	TestData
Read List	√		
Load Images	√	√	
Record Data			√

The Users data structure is only used at the first step of the entire test.

Table 15. Database



4.2. Java Class Diagram

Here we will just list the most important class.

Table 16. Java Class Diagram

Test
-Timer : java.util.Timer -StickReader : java.util.TimerTask
-showPic() -moveNextImg() -loadImg() -startTimer() -endTimer() +setPath()

This class is the core class of the Tester Terminal.

Class function:

Load images from the local disk.

Read the joystick position data from the DLL.

Calculate reaction time.

Class parameters:

Timer: a java.util.Timer object. This timer control the whole testing process. The task of the process repeats every 6 seconds. After all the tasks are finished, the timer will be deleted.

StickReader: this is a java inner class which extend from the java.util.TimerTask. This class will run in the timer. The task first loads the image and display on the screen. Then read the position data of the stick. If the data is valid, task will save it into the database.

Class methods:

Showpic: display images on the label.

moveNextImg: find the next image in the sequence.

setPath: After the user select the list, Transfer the list's ID to the application. This method will download images' data and generate files on the local disk.

loadImg:

startTimer: start an internal timer.

endTimer: stop an internal timer. Calculate the reaction time.

5. Discussion

5.1. Approach-Avoidance Meter test

We can measure people's evaluation of objects (such as games) by means of behaviour. More particularly, we can ask people to push or pull a joystick when seeing a stimulus (cf. Schnabel, Banse & Asendorpf, 2006; Chen et al., 1999; Newhagen, 1998). The longest reaction times for 'avoidance' behaviour (=pushing the joystick) should occur with the exposure of positively evaluated stimuli and the shortest reaction times for 'avoidance' behaviour (=pushing the joystick) should occur with the exposure of negatively evaluated stimuli. [7]

If the joystick meter works properly as a measurement of fun in games (cf. theoretical & methodological information above), then the reaction times of the positive images (such as the ice cream image and the birthday party image) should be inversely proportional to the negative images. We can test this statistically by analyzing whether the correlations between the positive and negative images are inversely proportional.

5.2. Final discussion

In general, the first preliminary tests with the joystick meter did not reveal any prove of this method working well to measure fun in games. The reasons for this do not lie in the technical measurement tool, but are more likely to be due to the characteristics of the test participants. We tested the joystick meter with very young children, who were apparently cognitively not yet sufficiently developed to use it in an appropriate way and to see the joystick meter as a meaningful tool. Because more than half of our test participants were too young to use the joystick meter –which we did not anticipate- we did not have a sufficient number of successful testing so that it became hard to draw any definite conclusions.
Further work should definitely test the joystick meter in more detail with a larger number of children who are older than six years old.

As for the technical set-up, the tool worked well. It only crashed 2 times out of the 55 times that we used it with all our participants. It was a handy tool to select any participant (or 'tester') and export his/her results to Excel, which allowed the researcher to quickly analyze the results in other programs such as Excel and spss. The tool also gave good feedback to the child users, each time they reacted with the joystick since an image with 'thumbs up' appeared. The time set between the exposures of two following images was perfectly: not too long neither too short.

The joystick meter is made as a generic tool that it can be used in many research settings for various target groups. The joystick meter is easy to use, both for the test participants as for the researcher. It registers all the information that the researcher might need in his/her research project such as the date of testing and the time, the direction and reaction time of the joystick, the image (order), the age, gender and name of the test participant. The researcher can easily select and upload the images that he/she is interested in to the tool. The

image lists can easily be adapted or deleted. So the system is quiet flexible.

Anyway, how to measure fun of a game is a science research with both theory and practice. Approach-Avoidance Meter is just a behavioural measurement software tool with hardware component to assist the research. And it always need to do some improvements which depend on the requirement of the research.

References

1. Schnabel, K., Banse, R., & Asendorpf, J. (2006). Employing Automatic Approach and Avoidance Tendencies for the Assessment of Implicit Personality Self-Concept. The Implicit Association Procedure (IAP). Experimental Psychology, 53(1), pp. 69-76.
2. http://en.wikipedia.org/wiki/Scrum_%28development%29
3. http://msdn.microsoft.com/en-us/library/aa987796%28VS.80%29.aspx
4. http://msdn.microsoft.com/zh-cn/library/z54t9z5f.aspx
5. Colman, A. (2001). *A Dictionary of Psychology*. Retrieved February 28, 2009.
6. http://www.humanbenchmark.com/tests/reactiontime/stats.php
7. Newhagen, J.E. (1998). TV news images that induce anger, fear, and disgust: Effects on approach-avoidance and memory. Journal of Broadcasting & Electroni